York University

AS/AK/ITEC 1620 3.0 – Section A

OBJECT-BASED PROGRAMMING

Summer 2002

Final Exam

Examiner: S.Y. Chen
Duration: Two Hours

This exam is closed textbook(s) and closed notes. Use of any electronic device (e.g. for computing and/or communicating) is NOT permitted.

Do not unstaple this test book – any detached sheets will be discarded. Answer all questions in the space provided. No additional sheets are permitted.

Work independently. The value of each part of each question is indicated. The total value of all questions is 100.

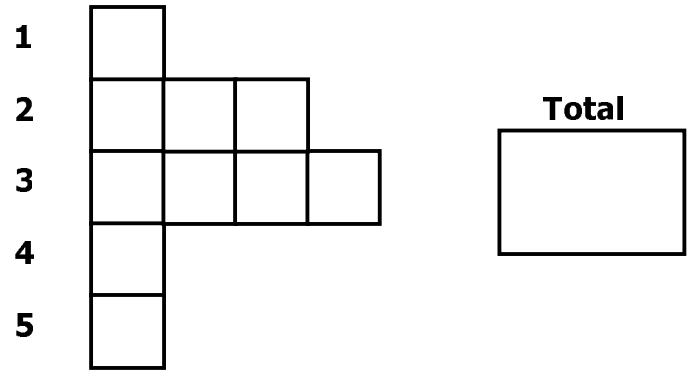
Write your name and student number in the space below. Do the same on the top of each sheet of this exam where indicated.

NOTE: YOU MAY USE PEN OR PENCIL.

Surname: _____

Given Names: _____

Student Number: _____



Question 1 (15 marks) Control Structures:

Write a static method in JAVA that will determine if the passed parameters represent a valid triangle. Each parameter represents the length of one side of the triangle, and the method returns `true` if the sides can form a valid triangle and `false` otherwise. A valid triangle has all sides greater than 0 in length, and each side must be shorter than the total length of the other two sides.

For example, the following statements would lead to the underlined output:

Example 1:

```
York.println( isValidTriangle( 3, 4, 7 ));
```

false

Example 2:

```
York.println( isValidTriangle( 3, 4, 5 ));
```

true

Example 3:

```
York.println( isValidTriangle( 3, 0, 5 ));
```

false

Example 4:

```
York.println( isValidTriangle( 10, 1, 10 ));
```

true

Example 5:

```
York.println( isValidTriangle( 8, 4, -1 ));
```

false

Please write your method on the following page.
You may use this page for rough work, but anything on this page will not be graded.

Surname:_____ First name:_____ Student #: _____

```
public static boolean isValidTriangle (int a, int b, int c)
{
  if ( a <= 0  ||   b <= 0    ||   c <= 0 )
     return false;
  else if ( a > b+c   ||   b > a+c    ||   c > a+b )
     return false;

  return true;









}
```

Question 2 (20 marks) Parameters, Scope, References:

Answer all three parts below.

The file `AnInt.java` contains the following implementation of the `AnInt` class:

```
public class AnInt
{
     public int data;

     public AnInt (int data)
     {
          this.data = data;
     }

     public void update(TwoInts other)
     {
          other.update(this);
     }
}
```

The file `TwoInts.java` contains the following implementation of the `TwoInts` class:

```
public class TwoInts
{
     public AnInt int1;
     public AnInt int2;

     public TwoInts (AnInt int1, AnInt int2)
     {
          this.int1 = int1;
          this.int2 = int2;
     }

     public void update(AnInt other)
     {
          other.data = int1.data + int2.data;
     }
}
```

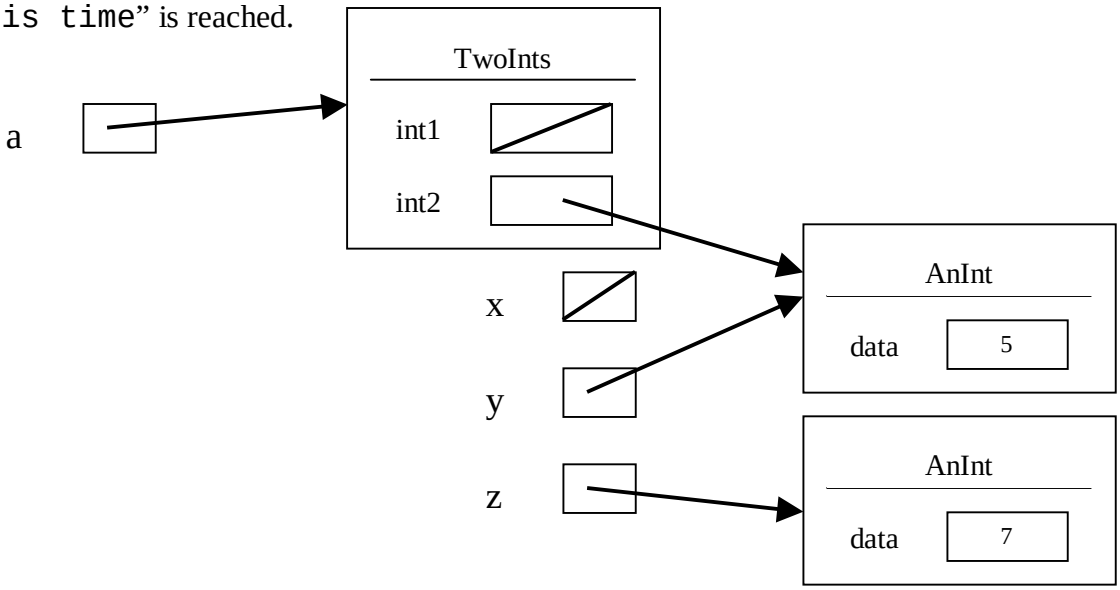
The `main` method in the file `MainClass.java` uses the above classes:

```
import york.*;
public class MainClass
{
     public static void main(String[] args)
     {
          AnInt x = null;
          AnInt y = new AnInt(5);
          AnInt z = new AnInt(7);
          TwoInts a = new TwoInts(x, y);

          // Part 1 – draw the object diagrams at this time

          x = new AnInt(3);
          y = z;
          z.data = 8;
          z = new AnInt(4);
          a.int1 = z;

          // Part 2 – draw the object diagrams at this time

          x.update(a);

          York.println(“” + a.int1.data + a.int2.data +
               x.data + y.data + z.data);

          // Part 3 – write the output of the above line

     }
}
```

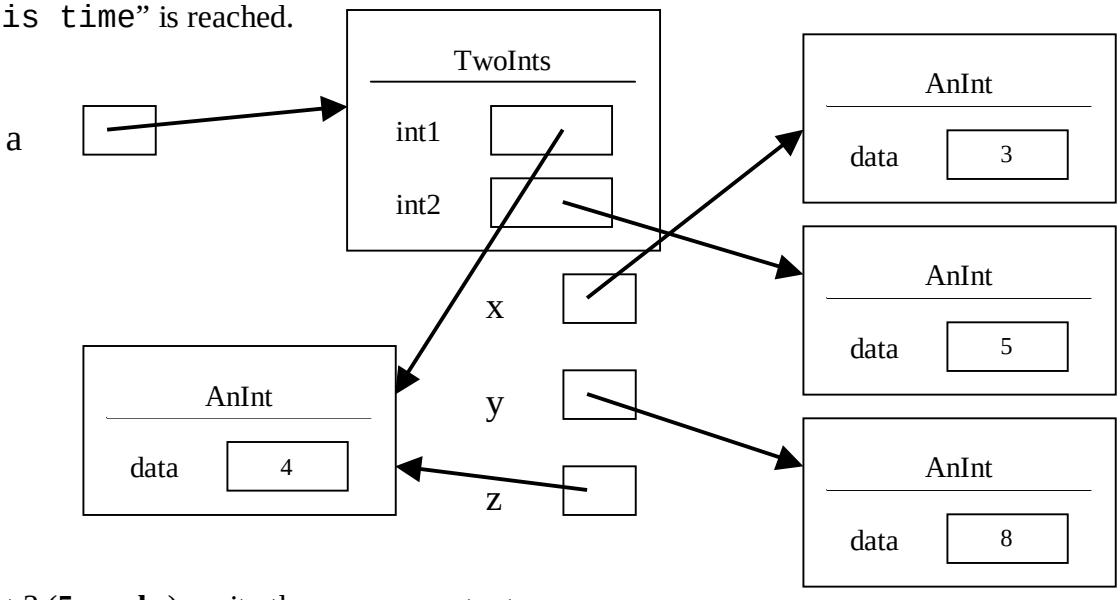
Surname:_____ First name:_____ Student #: _____

When `java MainClass` is executed,

Part 1 (**5 marks**): draw the object diagrams for all identifiers of `AnInt` and `TwoInts` when the comment line "`// Part 1 – draw the object diagrams at this time`" is reached.



Part 2 (**10 marks**): draw the object diagrams for all identifiers of `AnInt` and `TwoInts` when the comment line "`// Part 2 – draw the object diagrams at this time`" is reached.



Part 3 (**5 marks**): write the program output.

```
45984
```

Question 3 (30 marks) Building Classes:

Answer all four parts below.

The API for the `CreditCard` class is given below. Each instance of this class represents a credit account with a certain credit limit for a currency that has only whole units (e.g. Lira or Yen). The default credit limit is 10,000 and the penalty is 200 for attempting to make a charge that would cause the balance to exceed the credit limit. This class performs no error checking other than that specified in the API below.

Field Summary	
`static int`	`BASE_LIMIT` The default credit limit for a new credit card.
`static int`	`PENALTY` The penalty charge incurred for attempting to make a charge that would cause the balance to exceed the credit limit.
Constructor Summary	
`CreditCard()` Construct a credit card, and set the credit limit to the `BASE_LIMIT`.	
`CreditCard(int limit)` Construct a credit card, and set the credit limit to `limit`.	
Method Summary	
`boolean`	`makeCharge(int charge)` Attempts to make a charge to the credit account. If the new balance would exceed the credit limit, the charge is not processed – the balance is increased by the `PENALTY` amount, and the method returns `false`. Otherwise, the `charge` is added to the balance and the method returns `true`.
`void`	`makePayment(int payment)` Decreases the balance by the amount of the `payment`.
`void`	`raiseLimit(int amount)` Increases the credit limit by the specified `amount`.

Surname:_____ First name:_____ Student #: _____

Part 1 (**5 marks**) class and instance variables: List all of the class and instance variables (with appropriate modifiers) that you need to implement the `CreditCard` class.

```
public static final int BASE_LIMIT = 10000;
public static final int PENALTY = 200;

private int limit;
private int balance;
```

Part 2 (**5 marks**) constructors: Implement the one-parameter constructor.

```
public CreditCard(int limit)
{
  this.limit = limit;




}
```

Part 3 (**5 marks**) constructors: Employing reuse, implement the default constructor.

```
public CreditCard()
{
  this(BASE_LIMIT);




}
```

Surname:_____ First name:_____ Student #: _____

Part 4 (**15 marks**) methods: Implement the `makeCharge()` method.

```
public boolean makeCharge( int charge )
{
  if ( balance + charge > limit )
  {
     balance += PENALTY;
     return false;
  }

  balance += charge;
  return true;









}
```

Question 4 (20 marks) Using API's:

The API for the `Money` class is given below. Each instance of this class represents an amount of dollars and cents. The amount of cents will be an integer between 0 and 99 (inclusive).

Constructor Summary	
`Money()` Construct a Money amount with 0 dollars and 0 cents.	
`Money(int d, int c)` Construct a Money amount with `d` dollars and `c` cents.	
Method Summary	
`void`	`add(Money amount)` Increase this money amount by `amount`.
`boolean`	`isGreaterThan(Money amount)` Returns `true` if this money amount is greater than `amount`.
`void`	`subtract(Money amount)` Decrease this money amount by `amount`.

The API for the `TaxCalculator` class is given below. Each instance of this class can calculate the amount of `Money` that will be taxed for an `Item`. There are two types of taxes (e.g. GST and PST) and two different rates. These taxes are levied depending on the item type (e.g. `Item.FOOD`, `Item.PREPARED_FOOD`, or `Item.PRODUCT`).

Constructor Summary	
`TaxCalculator(int rateGST, int ratePST)` Construct a tax calculator with the GST set to `rateGST` % and the PST set to `ratePST` %.	
Method Summary	
`void`	`assignTaxes(int type, boolean t1, boolean t2)` Specifies that items of the given `type` should be assessed the GST if `t1` is `true` and the PST if `t2` is `true`.
`Money`	`calculateTax(Money price, int type)` Determines which taxes to apply for the given item `type`, multiplies the `price` by the appropriate tax rate(s), and returns the amount of tax that will be charged on the item.

The API for the `Item` class is given below. Each instance of this class represents an item with a name, type, and price.

Field Summary	
`static int`	`FOOD` The value that specifies that an item is of the "food" type (for the tax calculator).
`static int`	`PREPARED_FOOD` The value that specifies that an item is of the "prepared food" type (for the tax calculator).
`static int`	`PRODUCT` The value that specifies that an item is of the "product" type (for the tax calculator).
Constructor Summary	
`Item()` Construct an item of the "product" type without a name and a price of 0 dollars and 0 cents.	
`Item(String name, Money price, int type)` Construct an item with the specified `name`, `price`, and `type`.	
Method Summary	
`String`	`getName()` Returns the name for this item.
`Money`	`getPrice()` Returns the price for this item.
`int`	`getType()` Returns the type for this item.
`void`	`setName(String name)` Sets the name of this item to the given `name`.
`void`	`setPrice(Money price)` Sets the price of this item to the given `price`.
`void`	`setType(int type)` Sets the type of this item to the given `type`.

Write a static method in JAVA that will determine the total purchase price for the given item, tax calculator, and item quantity.

For example, a tax calculator for Ontario would have GST set to 7%, PST set to 8%, food items without GST and PST, prepared food items with GST only, and products with both GST and PST.

Using the above tax calculator, buying two sandwiches (prepared food items) with a price of 2 dollars and 50 cents would then lead to the following result:

Money totalPrice = calculateTotalPrice(sandwich, ontarioTaxCalculator, 2);

totalPrice → 5 dollars and 35 cents

Buying four pens (product items) with a price of 0 dollars and 75 cents would then lead to the following result:

Money totalPrice = calculateTotalPrice(pen, ontarioTaxCalculator, 4);

totalPrice → 3 dollars and 45 cents

Please write your method on the following page.
You may use this page for rough work, but anything on this page will not be graded.

Surname:_____ First name:_____ Student #: _____

```
public static Money calculateTotalPrice
     (Item item, TaxCalculator taxCalc, int quantity)
{
  Money total = new Money();
  Money price = item.getPrice();
  Money tax = taxCalc.calculateTax(price, item.getType());

  for (int i = 0; i < quantity; i++)
  {
     total.add(price);
     total.add(tax);
  }

  return total;









}
```

Question 5 (15 marks) Arrays:

Write a static method in JAVA that will convert a partially populated array of ints (with at least one element) into a fully populated array. Each element of the final array must have the value of the smallest element in the original array.

For example, the outcomes for the final array (shown underlined) of the following method calls (from the same class) are given below:

Example 1:

```
int[] array1 = new int[] {1, 2, 3, 5, 3, 2, 1};

makeMinimum(array1, array1.length);
```

array1 → [1 1 1 1 1 1 1]

Example 2:

```
int[] array2 = new int[] {12, 6, 10, 2, 7};

makeMinimum(array2, 3);
```

array2 → [6 6 6 6 6]

Please write your method on the following page.
You may use this page for rough work, but anything on this page will not be graded.

Surname:_____ First name:_____ Student #: _____

```
public static void makeMinimum(int[] ar, int count)
{
  int min = ar[0];

  for (int i = 1; i < count; i++)
  {
     if (ar[i] < min)
       min = ar[i];
  }

  for (int i = 0; i < ar.length; i++)
     ar[i] = min;









}
```